Câu 1

1.1 Actor của hệ thống

a. *Primary actor*

- Quản trị viên: là những người chịu trách nhiệm quản lý và duy trì hệ thống bán hàng trực tuyến BoMoC. Vai trò của họ bao gồm: Quản lý dữ liệu sản phẩm, Quản lý nhà cung cấp, Quản lý người dùng, Quản lý thanh toán,…

- Khách hàng: là những người dùng cuối cùng của hệ thống, truy cập trang web hoặc ứng dụng di động để thực hiện các hoạt động mua sắm. Khách hàng tìm kiếm, xem sản phẩm, thêm sản phẩm vào giỏ hàng, thực hiện thanh toán trực tuyến và quản lý tài khoản cá nhân. Họ tạo đơn đặt hàng, đánh giá sản phẩm, và cung cấp phản hồi về trải nghiệm mua sắm

b. *Secondary actor*

- Hệ thống thanh toán: Cho phép khách hàng chọn các phương thức thanh toán như thẻ tín dụng, chuyển khoản ngân hàng, ví điện tử, hoặc các phương thức thanh toán trực tuyến khác; Xử lý các giao dịch thanh toán
- Hệ thống thông báo: gửi thông báo và cập nhật đến khách hàng và các bên liên quan về các sự kiện quan trọng và hoạt động trong hệ thống qua email hoặc qua ứng dụng

1.2. Chức năng cho Admin

Đây là những chức năng được thực hiện bởi người dùng quản trị.

- Đăng nhập dành cho admin
- Quên mật khẩu dành cho admin
- Chỉnh sửa hồ sơ cho admin
- Đổi mật khẩu cho admin
- Chức năng đăng xuất
- **Quản lý khách hàng**
  + Thêm khách hàng mới
  + Chỉnh sửa khách hàng
  + Xem thông tin chi tiết của Khách hàng
  + Danh sách tất cả khách hàng
- **Quản lý sách**
  + Thêm sách mới
  + Chỉnh sửa sách đã có

+ Xem chi tiết sách
+ Danh sách tất cả sách

- **Quản lý clothes**
  + Thêm mặt hàng quần áo mới
  + Chỉnh sửa thông tin mặt hàng quần áo
  + Xem thông tin chi tiết của mặt hàng quần áo
  + Danh sách tất cả mặt hàng quần áo
- **Quản lý điện thoại**
  + Thêm điện thoại mới
  + Chỉnh sửa điện thoại đã có trên hệ thống
  + Xem thông tin chi tiết của điện thoại
  + Danh sách tất cả điện thoại
- **Quản lý danh mục sách**
  + Thêm danh mục sách mới
  + Chỉnh sửa danh mục sách đang thoát
  + Xem chi tiết Danh mục sách
  + Danh sách tất cả các loại sách
- **Quản lý danh mục quần áo**
  + Thêm danh mục quần áo mới
  + Chỉnh sửa danh mục quần áo đang thoát
  + Xem chi tiết Danh mục quần áo
  + Danh sách tất cả các loại quần áo
- **Quản lý danh mục điện thoại**
  + Thêm danh mục điện thoại mới
  + Chỉnh sửa danh mục điện thoại đang thoát
  + Xem chi tiết Danh mục điện thoại
  + Danh sách tất cả các loại điện thoại
- **Quản lý đơn hàng**
  + Tạo đơn hàng mới
  + Chỉnh sửa đơn hàng đã có
  + Xem thông tin chi tiết của đơn hàng
  + Danh sách tất cả các đơn hàng

1.3. Chức năng cho Customer

Chức năng được thực hiện bởi người dùng Khách hàng:

- Đăng ký: Khách hàng có thể đăng ký trên website bằng số điện thoại và email.
- Đăng nhập khách hàng : Đây là form đăng nhập, từ đó khách hàng có thể đăng nhập vào hệ thống

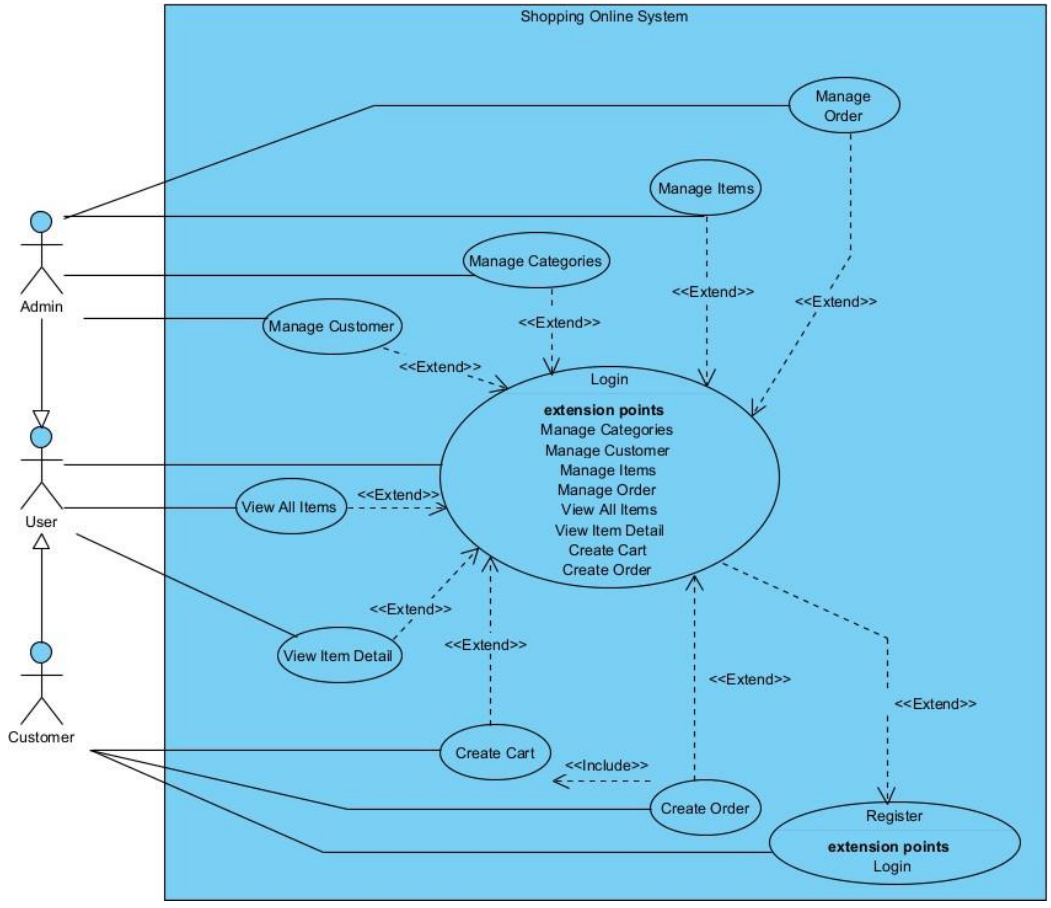
- Customer Cart: Đây là giỏ hàng của khách hàng.
  + Thêm sản phẩm vào giỏ hàng
  + Chỉnh sửa giỏ hàng
  + Xem giỏ hàng
- Customer Order: Đơn hàng của khách hàng

  Tạo đơn hàng mới: Chọn hình thức vận chuyển, Chọn hình thức thang toán
- Phản hồi cho sản phẩm đã mua: Đây là mẫu phản hồi của khách hàng để khách hàng có thể đưa ra những phản hồi về sản phẩm sau khi đã nhận được hàng
- Đổi mật khẩu: Đây là module đổi mật khẩu để khách hàng đổi mật khẩu tài khoản của mình.
- Xem danh sách tất cả các sản phẩm
- Tìm kiếm sản phẩm

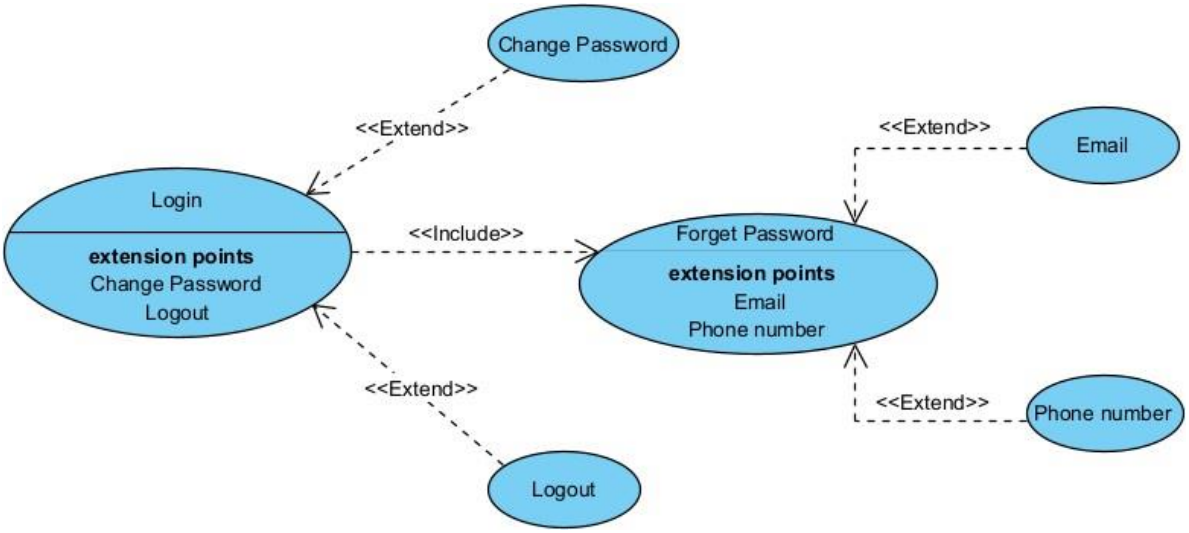
Câu 2

Use case diagram

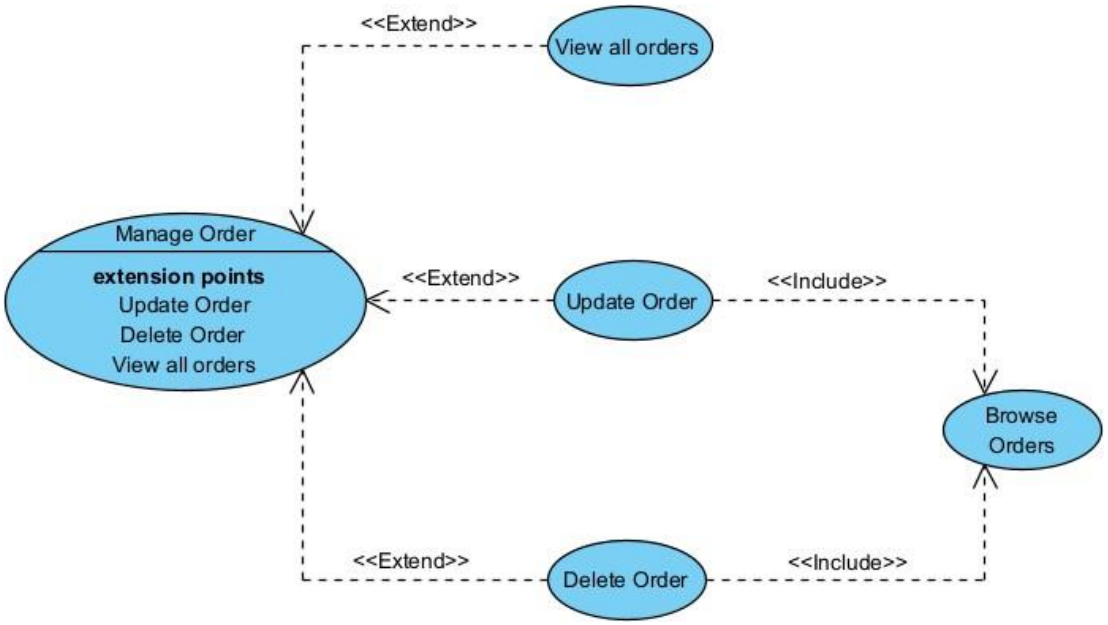
Use case tổng quát



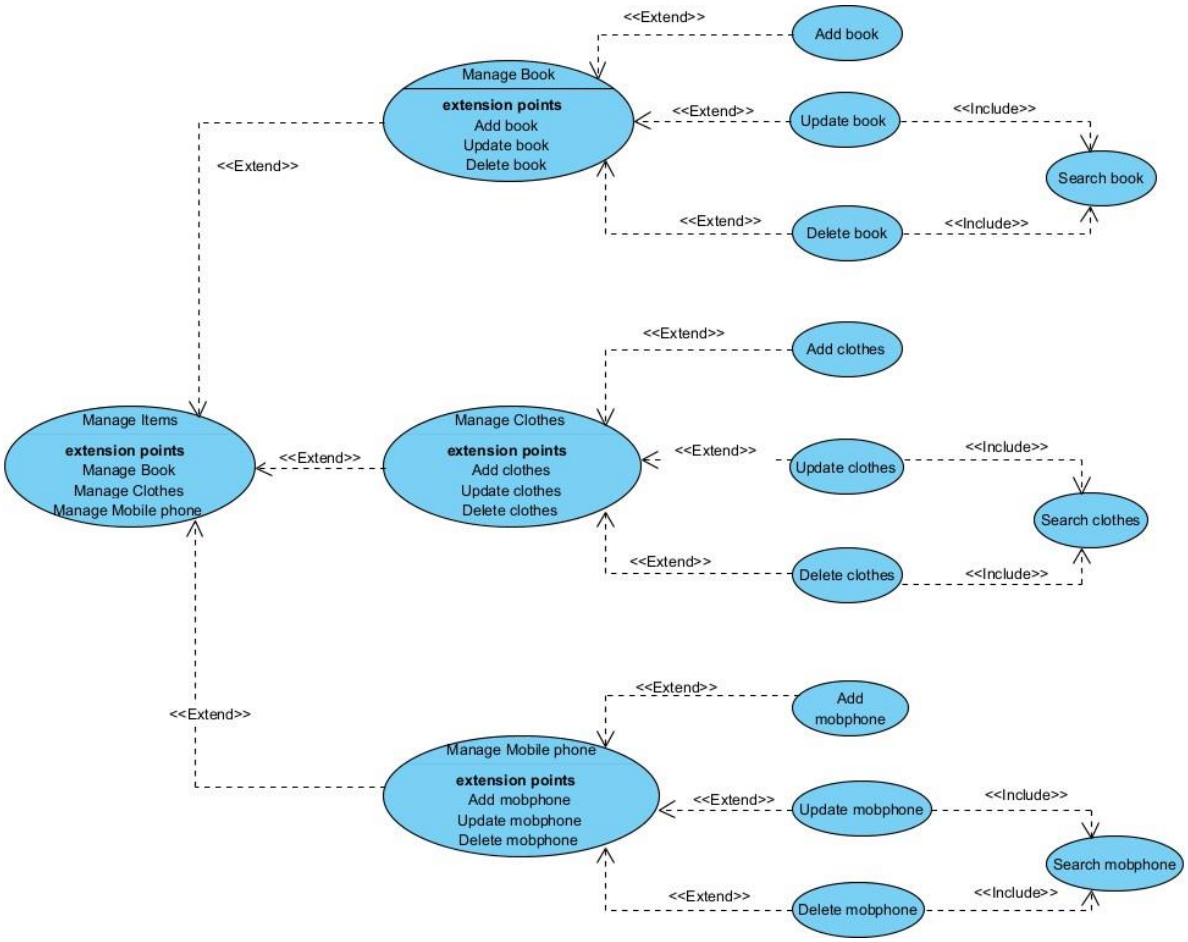
Use case login



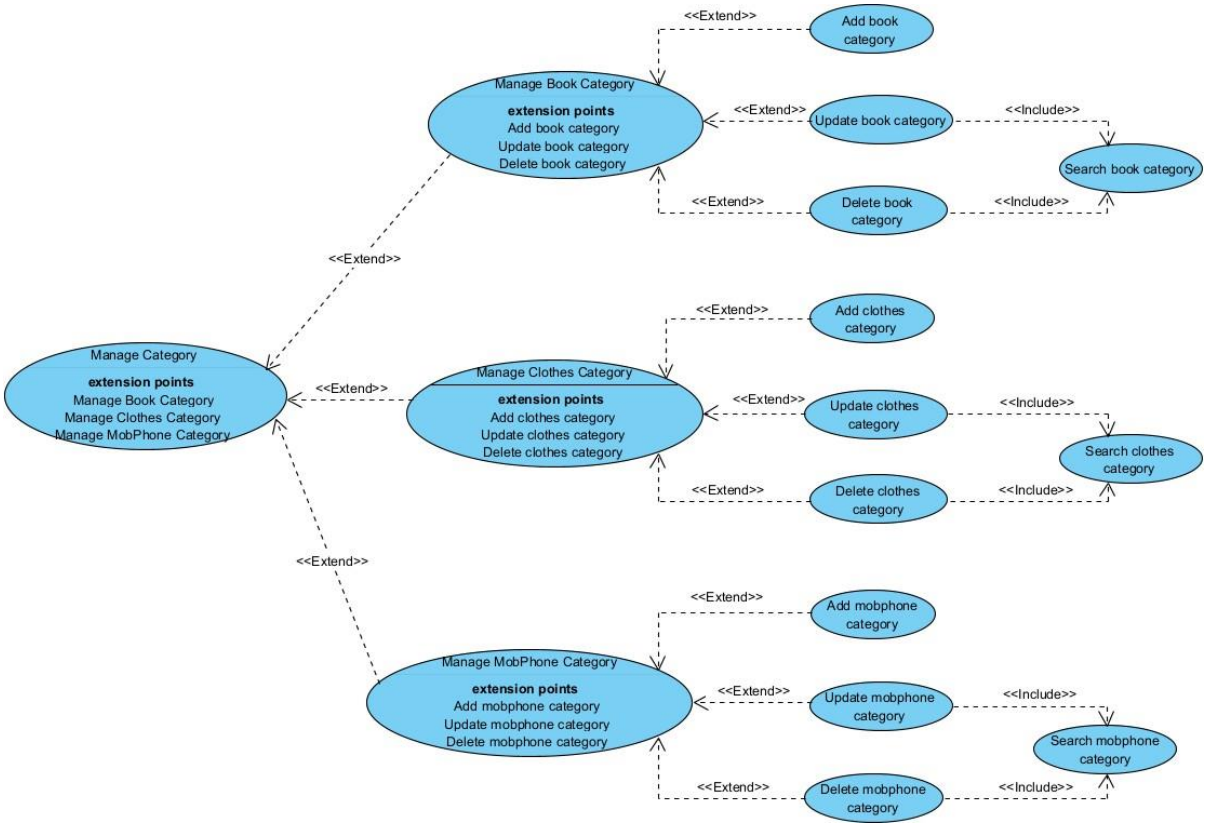
Use case Manage Order



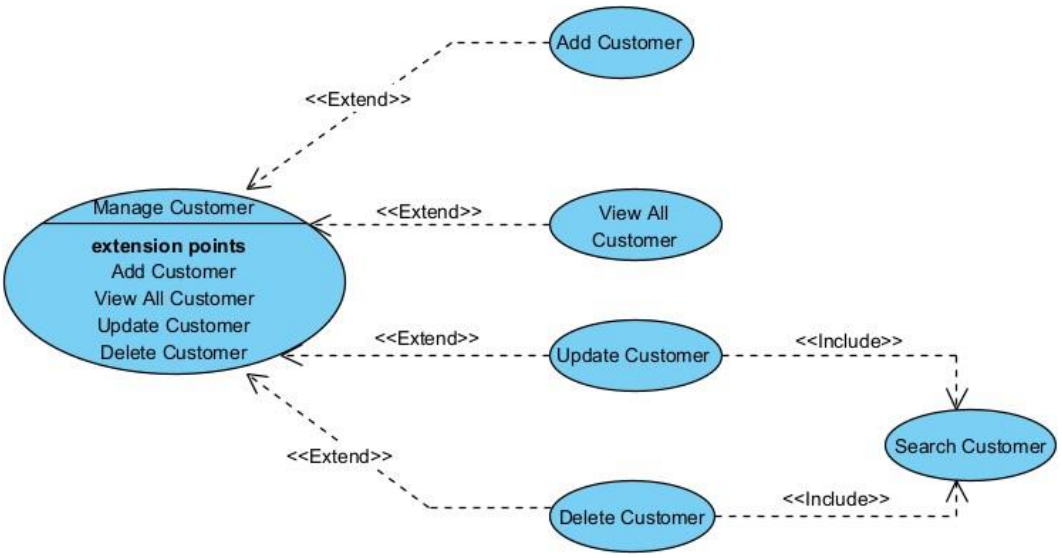
Use case Manage Items



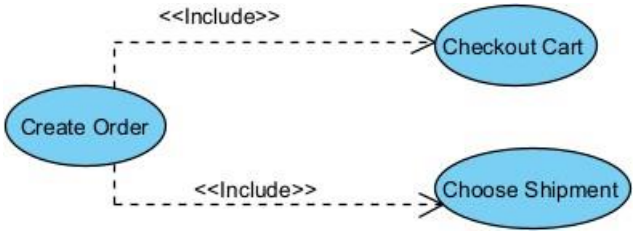
Use case Manage Category



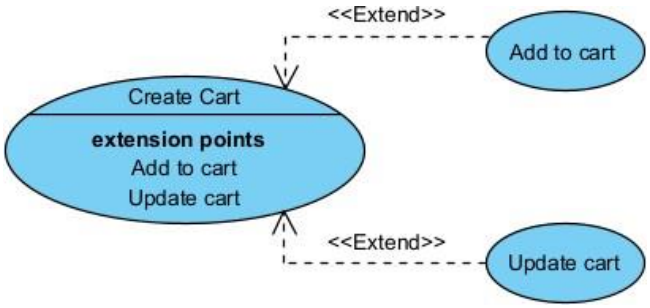
Use case Manage Customer



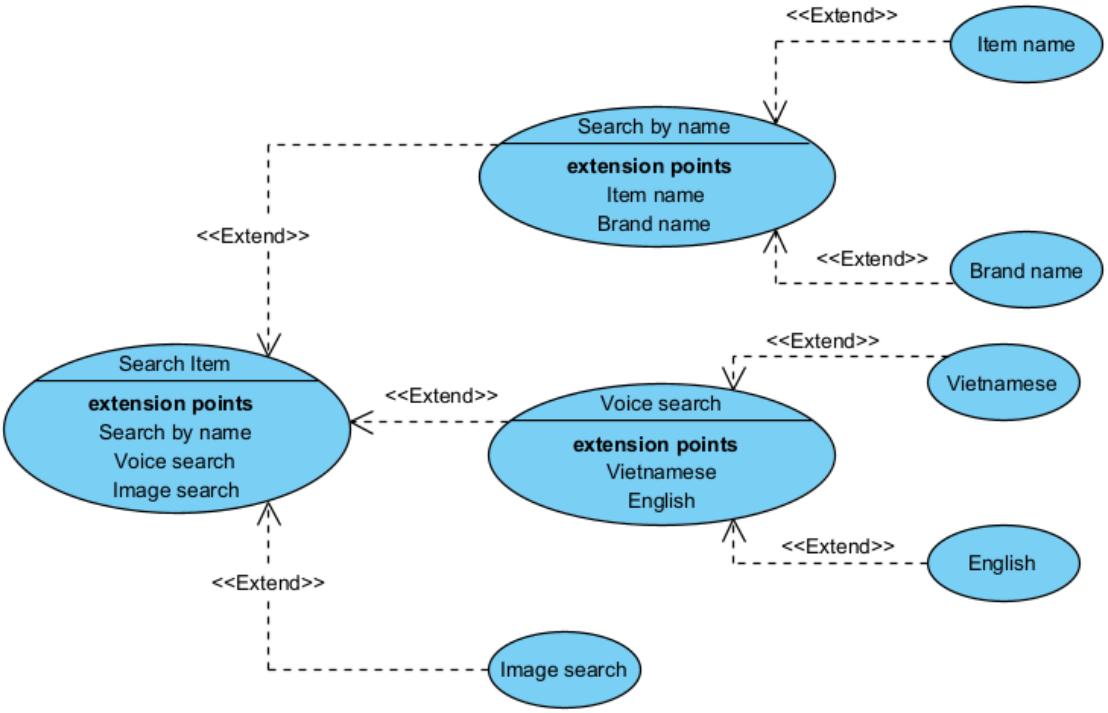
Use case Create order



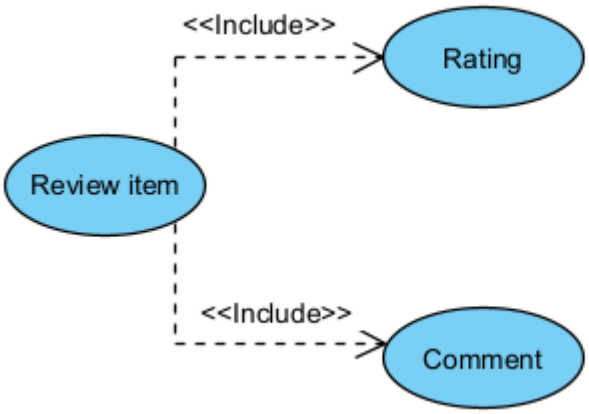
Use case Create Cart



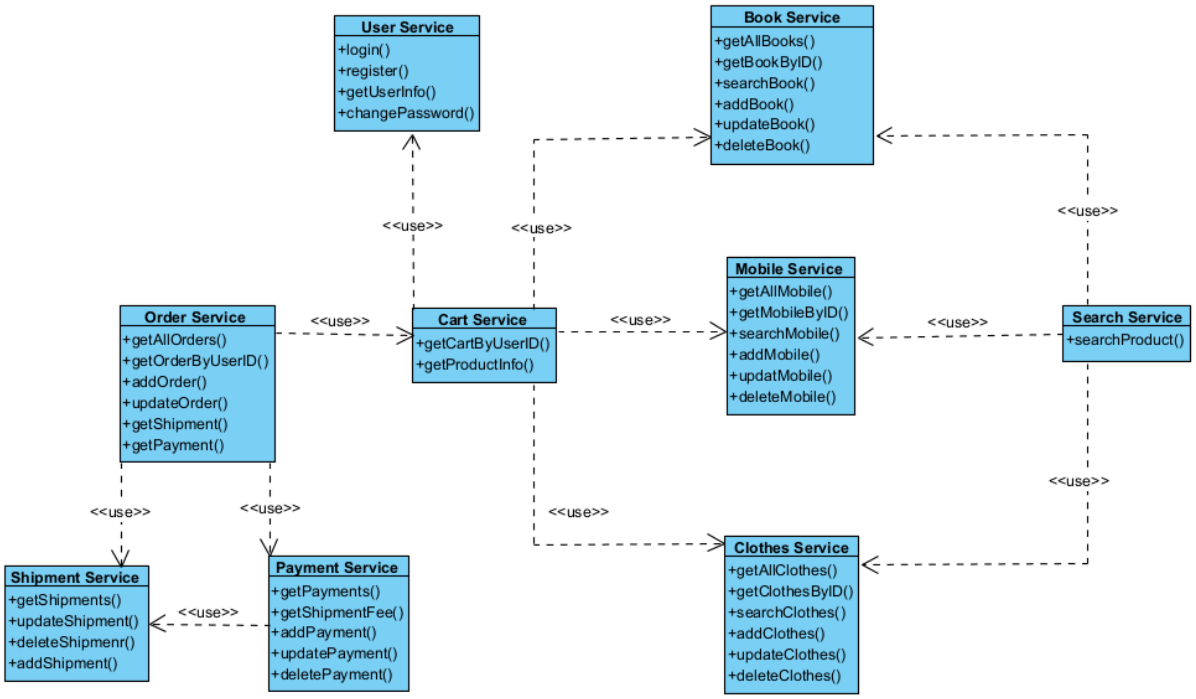
Usecase Search Item



Use case Review Item



Câu 3:



Câu 4:

Synchronous communication (Giao tiếp đồng bộ) và Asynchronous communication (Giao tiếp không đồng bộ) là gì?

Trong giao tiếp đồng bộ, nhiều bên liên tục lắng nghe và hành động dựa trên các phản hồi từ nhau. Một cách để hình dung khái niệm của giao tiếp đồng bộ là tưởng tượng một hệ thống trò chuyện trực tuyến thời gian thực được thiết kế cho hỗ trợ khách hàng của một nhà bán lẻ. Chuyên gia hỗ trợ nhanh chóng trao đổi tin nhắn với khách hàng để giúp theo dõi một đơn hàng, báo cáo về một giao hàng bị thiếu hoặc hỏi về một sản phẩm cụ thể.

Trong kịch bản này, cả người gửi và người nhận thiết lập một phiên giao tiếp. Khi phiên đã được thiết lập, cuộc trò chuyện hai chiều diễn ra mà không có hạn chế về việc ai nhập thông tin vào khi nào. Khi một bên gõ và gửi một tin nhắn trò chuyện, bên ở đầu kia có mặt và chờ đợi một cách tích cực để nhận và phản hồi tin nhắn. Đây chính là điều xác định giao tiếp đồng bộ.

Trong giao tiếp không đồng bộ, các bên không lắng nghe hoạt động để chờ tin nhắn. Dựa trên ví dụ ở trên, hãy tưởng tượng khách hàng sử dụng kênh hỗ trợ qua email thay vì trò chuyện trực tiếp. Truyền dẫn không đồng bộ xảy ra khi email được gửi đến bộ phận hỗ trợ của nhà sản xuất. Khách hàng không mong đợi nhận được một phản hồi trong thời gian thực. Thay vào đó, email đến tới nhà bán lẻ và nhân viên chọn lúc nào để đọc hoặc trả lời email.

Giao tiếp không đồng bộ thường gây ra một sự trễ giữa lúc người gửi khởi đầu tin nhắn và lúc người nhận phản hồi. Độ trễ này phụ thuộc vào phương tiện truyền thông. Một ví

dụ tương tự dựa trên thư tín vật lý có thể mất thêm thời gian khi vận chuyển. Hai bên tham gia giao tiếp không đồng bộ không làm việc cùng nhau trong thời gian thực. Trên thực tế, bất kỳ bên nhận nào cũng có thể hoàn toàn không biết họ đang tương tác với ai chính xác.

Hai hình thức truyền dữ liệu này có thể dễ hiểu trong giao tiếp giữa con người, nhưng nó khó khăn hơn đáng kể đối với chúng ta khi áp dụng chúng trong thiết kế phần mềm. Và trong khi các nguyên tắc cơ bản hướng dẫn việc thực thi mã chương trình, thời gian mà các truyền thông dựa trên phần mềm này cần để truyền thường có thể được đo bằng mili giây

Giao tiếp đồng bộ và không đồng bộ hoạt động thế nào?

Các ứng dụng tạo ra các thông điệp dưới dạng các cuộc gọi đến các hàm, dịch vụ và API. Cách mà chúng ta thiết kế các giao tiếp này ảnh hưởng đến hiệu suất của ứng dụng, tiêu thụ tài nguyên và khả năng thực hiện các nhiệm vụ.

Khi một thành phần phần mềm giao tiếp đồng bộ, nó sẽ im lặng cho đến khi nhận được một cuộc gọi, phản hồi, giá trị hoặc truyền dữ liệu khác. Ví dụ, thực thi đồng bộ xảy ra trong mua sắm trực tuyến. Một người dùng quyết định mua một sản phẩm, và hệ thống tạo ra một truy vấn để xác định xem có hàng tồn kho không. Ứng dụng đợi một phản hồi trước khi bắt đầu quá trình thanh toán. Thiết kế đồng bộ này ngăn chặn sự không phù hợp giữa hàng tồn kho và doanh số bán hàng.

Ngược lại, giao tiếp không đồng bộ cho phép mã tiếp tục chạy sau khi nó đã tạo ra một cuộc gọi hoặc phản hồi. Giao tiếp không đồng bộ đặc biệt quan trọng cho việc báo cáo và cảnh báo, như một ứng dụng sản xuất giám sát nhiệt độ của một lò nung công nghiệp, liên tục truyền cập nhật trạng thái và tự động gửi cảnh báo.

Loại ứng dụng này không bao giờ nên dừng lại và chờ đợi phản hồi trước khi di chuyển đến hành động tiếp theo. Thay vào đó, việc giao tiếp một mình nên kích hoạt hoặc nhân viên hoặc một ứng dụng khác để thực hiện hành động. Ví dụ, ứng dụng có thể gửi các cập nhật nhiệt độ không đồng bộ trong suốt cả ngày nhưng cũng kích hoạt một chuỗi khắc phục sự cố mỗi khi nhiệt độ vượt quá hoặc giảm dưới mức chấp nhận được.

Ví dụ giao tiếp đồng bộ:

```python
import asyncio
import aiohttp

async def check_inventory(product):
    async with aiohttp.ClientSession() as session:
        async with session.get(f'http://yourinventoryapi.com/{product}') as response:
            data = await response.json()
            if data['available']:
                return True
            else:
                return False

async def add_to_cart(product):
    if await check_inventory(product):
        print(f"Sản phẩm '{product}' có sẵn trong kho. Đã thêm vào giỏ hàng.")
    else:
        print(f"Sản phẩm '{product}' không có sẵn trong kho hoặc đã hết hàng.")

async def main():
    product_name = "Áo sơ mi"
    await add_to_cart(product_name)

asyncio.run(main())
```

Ví dụ không đồng bộ:

Dưới đây là một ví dụ về mã nguồn Python sử dụng thư viện requests để gửi yêu cầu HTTP đến một API mô phỏng việc cập nhật nhiệt độ và kích hoạt chuỗi khắc phục sự cố khi nhiệt độ vượt quá hoặc giảm dưới mức chấp nhận được.

```python
import requests
import time

# Hàm gửi yêu cầu cập nhật nhiệt độ
def update_temperature(new_temperature):
    # Gửi yêu cầu POST đến API để cập nhật nhiệt độ
    response = requests.post('https://your-api-endpoint.com/update_temperature', json={'temperature': new_temperature})
    if response.status_code == 200:
        print("Đã cập nhật nhiệt độ mới:", new_temperature)
    else:
        print("Lỗi khi cập nhật nhiệt độ")

# Hàm kích hoạt chuỗi khắc phục sự cố
def activate_maintenance():
    # Gửi yêu cầu POST đến API để kích hoạt chuỗi khắc phục sự cố
    response = requests.post('https://your-api-endpoint.com/activate_maintenance')
    if response.status_code == 200:
        print("Chuỗi khắc phục sự cố đã được kích hoạt")
    else:
        print("Lỗi khi kích hoạt chuỗi khắc phục sự cố")

# Hàm kiểm tra nhiệt độ và thực hiện các hành động cần thiết
def check_temperature():
    # Giả sử bạn có một API để lấy nhiệt độ hiện tại
    current_temperature = requests.get('https://your-api-endpoint.com/get_current_temperature').json()
```

```
    # Kiểm tra nhiệt độ và thực hiện hành động phù hợp
    if current_temperature > upper_threshold:
        print("Nhiệt độ quá cao! Kích hoạt chuỗi khắc phục sự cố.")
        activate_maintenance()
    elif current_temperature < lower_threshold:
        print("Nhiệt độ quá thấp! Kích hoạt chuỗi khắc phục sự cố.")
        activate_maintenance()
    else:
        print("Nhiệt độ trong phạm vi chấp nhận được.")

# Thiết lập ngưỡng nhiệt độ
upper_threshold = 30  # Đổi giá trị ngưỡng nhiệt độ tùy thuộc vào yêu cầu cụ thể
lower_threshold = 10  # Đổi giá trị ngưỡng nhiệt độ tùy thuộc vào yêu cầu cụ thể

# Vòng lặp chính
while True:
    check_temperature()
    time.sleep(60)  # Kiểm tra nhiệt độ mỗi phút
```

Câu 5

Giao tiếp đồng bộ:

- Payment service: cần đợi người dùng xác nhận thanh toán và xác thực được người dùng mới bắt đầu thanh toán. Và cần đợi hệ thống xử lý thanh toán xong mới có thể lấy được trạng thái thanh toán và update trạng thái thanh toán

- Search service: cần đợi các api search book của service book, search clothes của service clothes, search mobile của service mobile trả về kết quả mới có sản phẩm hiển thị lên cho người dùng

- User service:

  Khi user đăng nhập, sau khi gọi api login, cần đợi trả về kết quả mới quyết định báo lỗi hay thông báo đăng nhập thành công và hiển thị số lượng sản phẩm trong giỏ hàng

- Book service: khi thêm sách mới, sau khi gọi api để thêm sách, cần đợi kết quả trả về rồi quyết định hiển thị thông báo lỗi thêm sách hay thêm sách thành công. Tìm sách cũng tương tự như vậy, xem chi tiết sách cũng tương tự như vậy.

- Tương tự với Clothes service và Mobile Service

- Cart service:

  + Khi gọi api của book, clothes, mobile để lấy thông tin sản phẩm thì cần đợi các api này trả về kết quả rồi mới có thể hiển thị lên giao diện của người dùng

Giao tiếp không đồng bộ:

- Order service: Sau khi người dùng đặt hàng, đơn hàng sẽ được chuyển sang cho người bán xác nhận. Lúc này người dùng có thể tiếp tục thực hiện các hành động khác mà không cần đợi người bán xác nhận đơn hàng